(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 843 253 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
20.05.1998 Bulletin 1998/21

(51) Int. Cl.^{6}: **G06F 9/302**, G06F 9/318, H03M 7/30

(21) Application number: **96830585.4**

(22) Date of filing: **15.11.1996**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV RO SI

(71) Applicant:
SGS-THOMSON MICROELECTRONICS s.r.l.
20041 Agrate Brianza (Milano) (IT)

(72) Inventors:
- **Costa, Raffaele**
  20154 Milano (IT)
- **Santinoli, Davide**
  20142 Milano (IT)

(74) Representative:
Pezzoli, Ennio et al
Jacobacci & Perani S.p.A.
Via Visconti di Modrone 7
20122 Milano (IT)

(54) A method for reducing the number of bits needed for the representation of constant values in a data processing device

(57) A method for reducing the number of bits needed to represent constant values in a data processing device (100) including the steps of defining a group of constant values by selecting them as a function of their statistical frequency of use, representing each constant value of this group in the instructions by means of a shorter coded operand field, loading a current instruction from a bus (105) in an instruction register (145), deriving a corresponding operand field from the coded operand field of the current instruction by expansion means (150), and selectively connecting the bus (105) and an output of the expansion means (150) as input to an arithmetic logic unit (110).

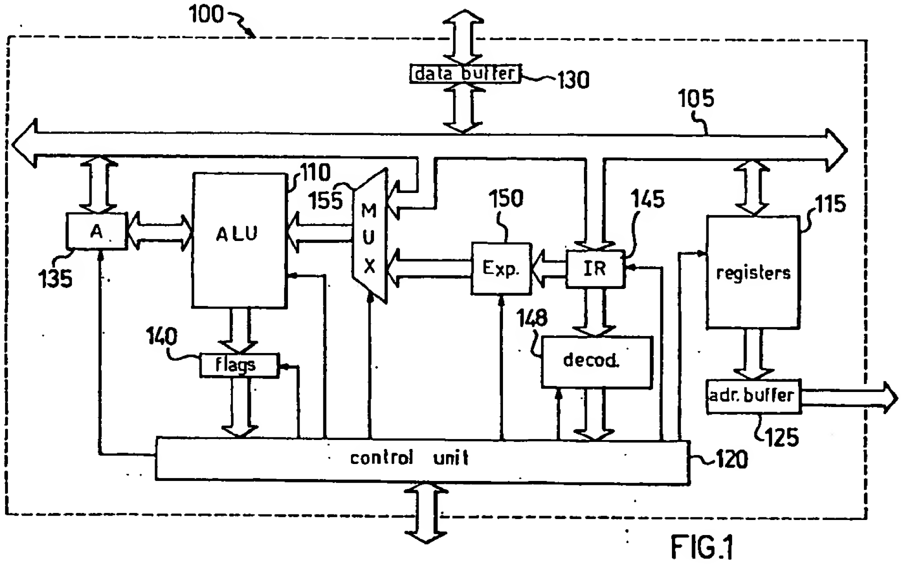

FIG.1

EP 0 843 253 A1

Printed by Xerox (UK) Business Services
2.16.3/3.4

Description

The present invention concerns a method and device for reducing the number of bits needed for the representation of constant values in a data processing device.

A data processing device, for example, a microprocessor, is constituted by an assembly of digital electronic circuits (hardware) able to execute instructions stored in a suitable program (software). A typical instruction comprises a sequence of binary digits (bits) and includes a part which indicates the type of operation to be executed (operation code), and a part which enables the data to be operated on (the operand field) to be identified. A data processing device generally utilises various addressing modes to access this data. In particular, in immediate addressing the data is specified by referring directly to its value in the instruction (immediate operand); this method of addressing is particularly useful when constant values are to be used. In the immediate addressing method, therefore, N bits of the operand field are needed to represent 2^N different constant values.

Various solutions are known for reducing the number of bits necessary for representing constant values in instructions intended to contain data to be addressed immediately. In so-called short constant systems, only Q bits (where $Q<N$) of lesser significance of the constant are represented in the operand field, obtaining the bits of greater significance thereof by sign extension or zero filling. In particular, in the case of sign extension, all the bits of the constant from the position Q+1 to the position N-1 are given the same value as the sign bit in position Q, while in the case of zero filling, these bits of the constant are given the value 0; the values which can be represented lie, respectively, in the range -2^{Q-1} to $2^{Q-1}-1$, and the range 0 to 2^Q-1. In single bit systems, the constants having a single bit with a value of 1 are represented by indicating the position of this bit in the constant; in this case, a number of bits P is necessary, given by the relationship $N = 2^P$, that is, $P = \log_2 N$. This system is usually used for instructions at the single bit level for clear, set, test, and toggle operations.

A disadvantage of the known systems is that only certain types of constant can be represented shortened; these systems therefore impose severe limitations both on the value of the representable constants and on their structure.

In addition, the set of reduced length constants represented in the known systems is fixed and is independent of the different operative requirements of the data processing device.

The aim of the present invention is to overcome the aforesaid disadvantages. To achieve this aim a method is provided as described in the first claim.

The method of the present invention enables different types and structures of constant to be represented as a function of the requirements of a wide variety of applications.

In addition, the representation of the constants in the method of the present invention depends on the use envisaged for the data processing device; this enables the device to be adapted to the applications in which it is used, consequently improving their performance in each individual operating condition.

Further characteristics and advantages of the method of the present invention will become clearer from the following description of a preferred embodiment, given by way of non-limitative example, with reference to the accompanying drawings, in which:

Figure 1 is a data processing device which may be used to perform the method of the present invention; and
Figure 2 is an embodiment of the expansion unit of Figure 1.

With reference now to the drawings, and with particular reference to Figure 1, a data processing device 100, specifically a microprocessor (µP), is shown; however, the present invention also lends itself to being put into practice utilising different data processing devices such as, for example, a logic controller. As usual, wide arrows indicate the passage of words, while small arrows indicate the passage of single bit commands. The data processing device 100 includes an internal data bus 105 to which is connected an arithmetic-logic unit (ALU) 110 which is able to execute calculations and checks; a plurality of internal registers 115 such as general purpose registers, a program counter, an instructions address register, a stack pointer and the like, are also connected to the bus 105. The data processing device also includes a control unit 120 which controls the different functions of the various components and their access to the internal data bus 105 by means of suitable pulses. An address buffer 125 connected to the internal registers 115, and a data buffer 130 connected to the internal data bus 105, are connected respectively to an external address bus and an external data bus (not shown in the drawings); the control unit 120 is connected directly to an external control bus (not shown in the drawings). An accumulator register (A) 135, for storing an operand and the result of the various operations performed by the ALU 110, is connected to the internal data bus 105 and the ALU 110. A state register 140, the bits (flags) of which are used as indicators of particular operating conditions, is connected to the ALU 110 and the control unit 120. An instruction register (IR) 145 is used to receive a current instruction to be executed, or at least a portion thereof comprising its operation code, from the internal data bus 105; this operation code is sent to an instruction decoder unit 148, as a function of the output from which the control unit 120 activates the various circuit elements to execute the current instruction.

Some of the instructions utilised in the data processing device 100 include a constant value which must be input to the arithmetic-logic unit 110 in the form of an N bit operand field $C_{N-1}...C_1C_0$. In the method of the present invention, a group of constant values having a high statistical frequency of use is determined in advance. This determination is made, for example, from an analysis of the type of applications envisaged for the data processing device, or by utilising appropriate benchmark programs. The above-determined group of constant values is coded appropriately such that each constant is represented in the instructions by a coded operand field having fewer than N bits.

The method of the present invention lends itself to being put into practice using various systems for coding these constant values. In a first embodiment, constants having N bits are represented by a coded operand field of $2+\log_2N$ bits in the form $a_1a_0b_{L-1}...b_1b_0$. The bits $b_{L-1}...b_1b_0$ (with $L=\log_2N$) comprise a position field and identify a bit position B (from 0 to N-1) in the operand field. The bits a_1a_0, which may take four different values (00, 01, 10 and 11), form a selector which defines the structure of the operand field. For example, the selector a_1a_0 indicates which bits of the operand field have the value 1, in particular:

00: only the bit in position B;
01: all of the bits except for the bit in position B;
10: all of the bits from the first position (0) to position B;
11: all of the bits from position B+1 to the last position (N-1).

In other words, the operand field is obtained from a string of N bits of value 0 by giving the bit in position B the value 1 when $a_1=0$, or by giving all of the bits from position 0 to position B the value 1 when $a_1=1$; when $a_0=0$ the value thus obtained is left unchanged, while when $a_0=1$ this value is bit-wise inverted. This coding system is utilised advantageously for constants having a single bit of value 1 or 0, for constants of value 0(00000000), ±1 (00000001, 11111110), ±2 (00000010, 11111101), ±3 (00000011, 11111100), ±4 (00000100, 11111011), ±8 (00001000, 11110111) and so on, used for loading, comparisons, rapid additions and subtractions (the constant -1 is also used as a mask in a toggle bit command to replace the one's complement operation), and also for constants of the type 11...100...0 and 00...0.11...1 used for sign or zero extension, and for the insertion or extraction of bit fields in a word.

In an alternative embodiment of the present invention, constants having N bits are represented by an operand field coded as $3+\log_2N$ bits in the form $s_0a_1a_0b_{L-1}...b_1b_0$, in which the bit s_0 comprises a selector bit which defines which bit of the operand field is considered as the last bit in the decoding process described above. When $s_0=0$, position N-1 is considered to be the last position of the operand field while when $s_0=1$, position N/2-1 is considered to be the last position of the operand field; in the latter case, the bit position B (from 0 to N/2-1)is represented in the coded operand field only by the bits $b_{L-2}...b_1b_0$, while the bit b_{L-1} represents the value to be attributed to the bits of the operand field in the position N/2 to the position N-1. Several examples of constants having 16 bits coded according to the coding system described above with $3+\log_216=3+4=7$ bits are given below:

s_0	a_1a_0	$b_3b_2b_1b_0$	operand field
0	1 1	0 1 1 1	1111111100000000
0	0 0	0 0 0 1	0000000000000010
1	1 1	0 1 0 1	0000000011000000
1	1 0	1 0 1 0	1111111100000111

This coding system allows constants having two non-contiguous blocks of bits of value 1 (or 0) to be represented. This is particularly advantageous in reduced instruction set computers, or RISC, which are only able to operate on the full range of their registers; in this case, the system described above enables operations to be performed on data which is shorter than a register.

Returning to the drawing, the data processing device 100 includes an expansion unit 150 which receives a coded operand field from the instruction register 145 as input, and produces the corresponding operand field as output. The output of the expansion unit 150 is connected to a first input of a multiplexer circuit 155, the second input of which is connected to the internal data bus 105. The multiplexer 155 transfers one of its two inputs to its output connected to the ALU 110 in dependence on an appropriate control signal applied to a selector input. In particular, a portion (address field) of the operation code of every instruction contains an indication of the kind of address utilised thereby (for example, immediate, direct, indirect, relative, from the register, indexed and the like). In the case of immediate addressing, an appropriate bit of this field enables immediate addressing without coding to be distinguished from coded immediate addressing. In the case of immediate addressing without coding, the control unit 120 sends a signal for reading the immediate operand from the bus 105 and therefore causes the multiplexer 155 to transfer the data on the bus 105 to the ALU 110. In the case of coded immediate addressing, the control unit 120 enables the expansion unit 150 and, at the same time, commands the multiplexer 155 to transfer the output thereof to the ALU 110. This data processing device therefore enables the use of shorter instructions which require fewer memory access steps (machine cycles) to fetch them; this reduces the memory space occupied by the programs and increases their speed of execution.

With reference now to Figure 2, an embodiment of the expansion unit 150 is illustrated which may be used in combination with a coding system in the form $a_1a_0b_{L-1}...b_1b_0$ described above. The expansion unit 150 receives the coded operand field $a_1a_0b_{L-1}...b_1b_0$ as input, and produces the corresponding operand field $c_{N-1}...c_1c_0$ as output. The expansion unit 150 includes a decoder 202 having L input terminals (for receiving the position field $b_{L-1}...b_1b_0$ as input) and N output terminals $u_{N-1}...u_1u_0$ in which, for every input combination, there corresponds a single output u_B of value 1 associated with the position B identified by the position field $b_{L-1}...b_1b_0$. A corresponding combinatorial network 205-225 is associated with each of the outputs $u_{N-1}...u_1u_0$ of the decoder 202. The structure of these combinatorial networks is illustrated in detail with reference to the block 215 (similar considerations apply to the other combinatorial networks). The combinatorial network 215 includes an input terminal connected to a corresponding output terminal u_2 of the decoder 202, and an output terminal 235 which provides a corresponding bit c_2 of the operand field. The combinatorial network 215 includes three further input terminals 240-250 connected to three corresponding output terminals of the preceding combinatorial network 218, and three further output terminals 255-265 connected to three corresponding input terminals of the following combinatorial network 210; the output terminals of the last combinatorial network 205 are not used, while the input terminals of the first combinatorial network 225 receive, respectively, the value a_1, the fixed value 0 (typically connecting this input terminal to a reference, or earth, terminal), and the value a_0. The terminals 240 and 250 are connected directly to the terminals 255 and 265 respectively. The terminals 240 and 245 are connected to the input terminals of an AND gate 270; the terminal 230 and the output terminal of the AND gate 270 are connected to the input terminals of an OR gate 275. The output terminal of the OR gate is connected to the terminal 260; the terminal 250 and the same output terminal of the OR gate 275 are connected to the input terminals of an XOR gate 280. The output terminal of the XOR gate 280 is connected to the terminal 235.

In order to describe the operation of the expansion unit 150, the position field $b_{L-1}...b_1b_0$ is assumed to have the value 2, so that all of the outputs $u_{N-1}...u_1u_0$ of the decoder 202 take the value 0 apart from the output u_2=1. If the bit a_1 has the value 0, all of the AND gates 270 will have an input of value 0, and their outputs will therefore always be 0. The outputs of the OR gates 275 will therefore be equal to the input values $u_{N-1}...u_1u_0$, that is, all having the value 0 with the exception of that corresponding to the output u_2, which has the value 1. If the bit a_1 has the value 1, the AND gate of the first combinatorial network 225, having an input of value 0, will have an output of value 0; the output of the OR gate of the same combinatorial network 225 (connected to the input of the AND gate of the second combinatorial network 220) will therefore be equal to the input value u_{N-1}, that is, 0. Continuing analogously, it is understood that the AND gates 270 up to the one corresponding to the output u_2 (inclusive) have an input of value 0, so that its output will always be 0; the output of the corresponding OR gates 275 will therefore be equal to the input value $u_{N-1}u_{N-2}...u_2$, that is, 0 for the inputs $u_{N-1}u_{N2}...u_3$, and 1 for the input u_2. Instead, both of the inputs of the AND gates following that corresponding to the output u_2 have a value 1, so that their output will always be equal to 1; the output of the corresponding OR gates will therefore always be equal to 1. Considering now the bit a_0, it may be observed that when the bit a_0 has a value of 0, the output of the XOR gates 280 (which represents a corresponding bit of the operand field $c_{N-1}...c_1c_0$) is equal to that of the corresponding OR gates 275; when the bit a_0 has a value of 1, the output of the XOR gates 280 is, instead, the opposite of that of the corresponding OR gates 275.

Claims

1. A method for reducing the number of bits necessary to represent constant values for use in a data processing device (100) able to execute instructions including an operand field which represents a constant value, the method comprising the steps of:

   defining a group of constant values by selecting them as a function of their statistical frequency of use;
   representing each constant value of the said group in the instructions by a coded operand field having fewer bits than the operand field;
   at least partially loading a current instruction into an instructions register (145) from a bus (105);
   if the current instruction contains a coded operand field, deriving a corresponding operand field from the coded operand field of the current instruction using expansion means (150); and
   selectively connecting the bus (105) and an output of the expansion means (150) as input to an arithmetic-logic unit (110).

2. A method according to Claim 1, in which the selective connection step is executed as a function of the value of an address field of the current instruction.

3. A method according to Claim 1 or 2, in which the operand field is constituted by N bits, and the coded operand field is constituted by a position field of $L=\log_2N$ bits indicative of a bit position B in the operand field, and by a 2-bit selector field, the step of deriving the operand field from the coded operand field comprising the steps of setting the bit in

position B of the operand field to the value 1 in correspondence with a first value of the selector field, all the bits with the exception of the bit in position B to the value 1 in correspondence with a second value of the selector field, all the bits from a first position to the position B-1 to the value 1 in correspondence with a third value of the selector field, or all of the bits from the position B+1 to a final position to the value 1 in correspondence with a fourth value of the selector field.

4. A method according to Claim 3, in which the coded operand field further includes a mode selector bit, the last position being set to the value N-1 in correspondence with a first value of the mode selector bit, the last position being set to the value N/2-1 in correspondence with a second value of the mode selector bit, the bits of the position field less one bit representing the position of bit B, the said one bit representing the value of the bits of the operand field from the position N\2 to the position N-1 in correspondence with the second value of the mode selector bit.

5. A data processing device (100) able to execute instructions including an operand field which represents a constant value, each constant value of a group of constant values chosen as a function of their statistical frequency of use being represented in the instructions by a coded operand field with fewer bits than the operand field, the data processing device (100) including:

   an instruction register (145) for at least partially loading a current instruction from a bus (105);
   expansion means (150) for deriving a corresponding operand field from the coded operand field of the current instruction;
   a multiplexer (155) for selectively connecting the bus (105) and an output of the expansion means (150) as input to an arithmetic-logic unit (110).

6. A data processing device (100) according to Claim 5, in which the operand field is constituted by N bits and the coded operand field is constituted by a position field of $L=\log_2 N$ bits and a selector field having 2 bits, the expansion means (150) comprising a decoder (202) having L input terminals for receiving the position field as input and N output terminals, and N combinatorial networks (205-225) each having first (240), second (245) and third (250) input terminals, and first (255), second (260) and third (265) output terminals connected, respectively, to corresponding output terminals of a preceding combinatorial network (218) and corresponding input terminals of a following combinatorial network (210), the first, second and third input terminals of a first combinatorial network (225) receiving as input, respectively, a first bit of the selector field, a value 0 and a second bit of the selector field, a fourth input terminal (230) connected to a corresponding output terminal of the decoder (202), and a fourth output terminal (235) for producing a corresponding bit of the operand field as output, the first (240) and third (250) input terminals being connected, respectively, to the first (255) and third (265) output terminals, the first (240) and second (245) input terminals being connected as inputs to an AND gate (270), the fourth input terminal (230) and an output terminal of the AND gate (270) being connected as input to an OR gate (275), an output terminal of the OR gate being connected to the second output terminal (260), the output terminal of the OR gate (275) and the third input terminal (250) being connected as input to an XOR gate (280), an output terminal of the XOR gate (280) being connected to the fourth output terminal (235).

7. A data processing device(100) according to Claim 5 or 6, in which the data processing device (100) is a microprocessor.

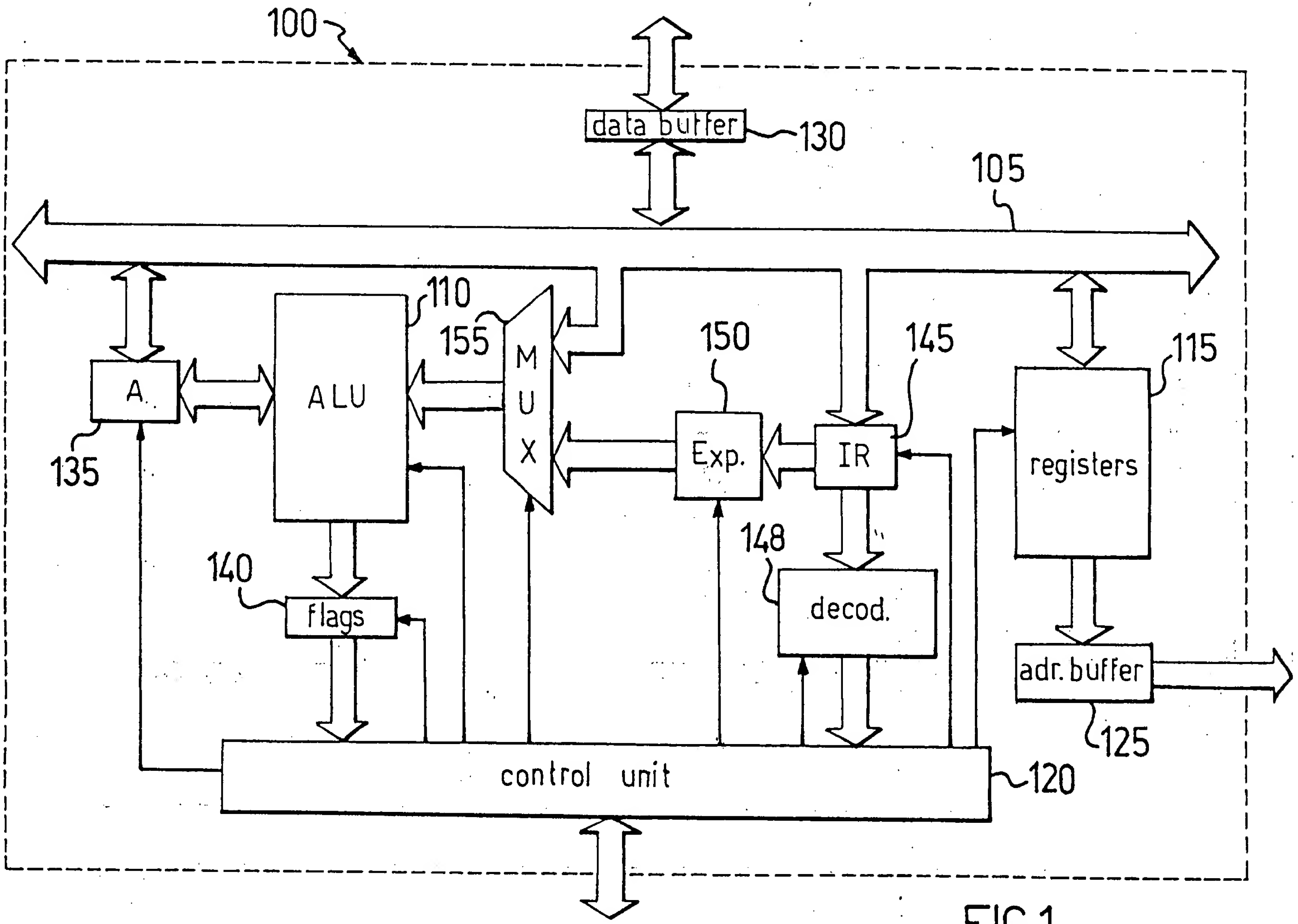
100
data buffer
130
105
110
155
M
U
X
150
145
115
A
ALU
Exp.
IR
registers
135
148
140
flags
decod.
adr. buffer
125
control unit
120

FIG.1

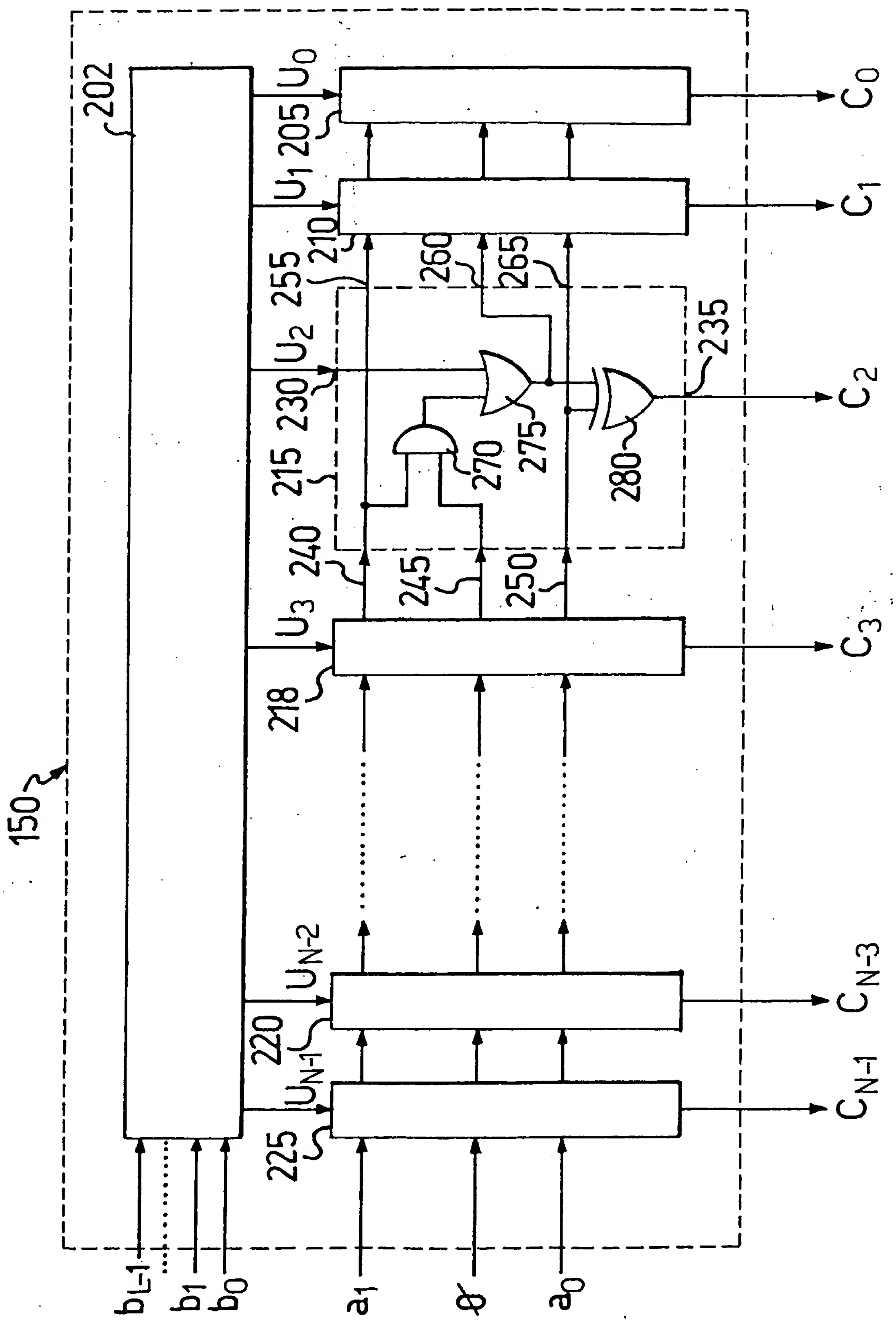
150
202
205
210
255
260
265
215
230
235
240
245
250
270
275
280
218
220
225
U0
U1
U2
U3
UN-2
UN-1
C0
C1
C2
C3
CN-3
CN-1
bL-1
b1
b0
a1
Ø
a0

FIG.2

European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 96 83 0585

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	EP 0 489 266 A (TOKYO SHIBAURA ELECTRIC CO) 10 June 1992	5	G06F9/302 G06F9/318 H03M7/30
A	* column 4, line 55 - column 5, line 30 * * column 11, line 25 - column 12, line 11 * * column 17, line 30 - column 18, line 40 * ---	1	
A	ELECTRONIC DESIGN, vol. 33, no. 2, January 1985, HASBROUCK HEIGHTS, NEW JERSEY US, pages 127-136, XP002029662 G. GOODHUE ET AL.: "Harvard rchitecture pushes microcontroller IC into high-speed realm" * page 129, left-hand column, paragraph 1 * ---	1	
A	US 4 192 010 A (KERNER WILLIAM R ET AL) 4 March 1980 ---		TECHNICAL FIELDS SEARCHED (Int.Cl.6)
A	IBM TECHNICAL DISCLOSURE BULLETIN, vol. 30, no. 5, October 1987, NEW YORK US, pages 79-80, XP000670477 "Padding a Microcode Control Word" * the whole document * -----	1	G06F H03M

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	17 April 1997	Daskalakis, T

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons
& : member of the same patent family, corresponding document